Dancing Link

$n + e$

Tsinghua University

2016 年 6 月 22 日

About

- 听起来很厉害的样子
- 一切都源于——≪ 靶型数独 ≫?

About

- DLX 的理念: 动态减少搜索中的状态数
- 用链表来减少对无效内存的访问

About

- DLX 的理念: 动态减少搜索中的状态数
- 用链表来减少对无效内存的访问
- 具体代码一搜一大堆
- 至今没写过 (捂脸跑)

精确覆盖问题

- 01 矩阵化, 选取一个行的集合, 使得集合中每一列都恰好包含一个 1
- 行: 所有位置 $*$ 所有可能情况
- 列: 各种约束条件
- 怎么填 1: 第 i 行的情况满足第 j 个条件约束, 则 $A[i][j] = 1$, 否则 $A[i][j] = 0$

N Queens

- 行: $N * N$
- 列: $N + N + 2N - 1 + 2N - 1 = 6N - 2$. N 行, N 列, $2N - 1$ 左斜线, $2N - 1$ 右斜线
- 矩阵的每行都有 4 个 1, 分别分布在行域, 列域, 左斜线域, 右斜线域

N Queens

- 行: $N * N$
- 列: $N + N + 2N - 1 + 2N - 1 = 6N - 2$. N 行, N 列, $2N - 1$ 左斜线, $2N - 1$ 右斜线
- 矩阵的每行都有 4 个 1, 分别分布在行域, 列域, 左斜线域, 右斜线域
- 在编程求解这个问题时, 需要做一点变通, 因为左斜线域, 右斜线域的列不可能被全部覆盖, 因此只需行域和列域被完全覆盖就算找到问题的一个解了.
- Tips: 可以调整列的顺序来加快搜索速度 (行列行列 $\cdots$)

Sudoku

- 行: $N*N*N$. 因为一共 $N*N$ 个小格, 每个小格有 N 中可能性 $(1-N)$, 每一种可能对应这一行.
- 列: $(N+N+N)*N+N*N$. 其中前面 3 个 N 分别代表着 N 行 N 列和 N 小块, 乘以 N 表示 N 中可能, 每种可能只可以选一个. $N*N$ 表示 $N*N$ 个小格, 限制每一个小格只可以放一个地方.
- 如果一个位置已经确定, 则只插入一行, 否则插入 $N*N$ 行, 代表 $N*N$ 种可能.

Puzzle

- 行: 拼图个数 * 每个拼图能够放置的位置数
- 列: 拼图个数 + 格子个数. 这两个都要精确覆盖, 也就是有且仅有一个 1
- NOI2005 智慧珠游戏

- 我都说了我不会写 DLX···

- 我都说了我不会写 DLX···
- 常见的搜索优化: 调整搜索序, 二进制压位, 多加剪枝, 估价 (还要多少步才能达到最终局面, 如果以最优解法达到最终局面的步数还是比当前最优解大就 cut), 迭代加深

- 我都说了我不会写 DLX···
- 常见的搜索优化: 调整搜索序, 二进制压位, 多加剪枝, 估价 (还要多少步才能达到最终局面, 如果以最优解法达到最终局面的步数还是比当前最优解大就 cut), 迭代加深
- 调整搜索序: DLX 是动态调整搜索序, 于是我就搞了一个静态调整搜索序
- 二进制压位: and, or, xor 来调整状态, 省去访问数组的时间, 比链表还快
- 剩下两个: IDA*, 待会儿会讲, 不过我没用到

- 我都说了我不会写 DLX···
- 常见的搜索优化: 调整搜索序, 二进制压位, 多加剪枝, 估价 (还要多少步才能达到最终局面, 如果以最优解法达到最终局面的步数还是比当前最优解大就 cut), 迭代加深
- 调整搜索序: DLX 是动态调整搜索序, 于是我就搞了一个静态调整搜索序
- 二进制压位: and, or, xor 来调整状态, 省去访问数组的时间, 比链表还快
- 剩下两个: IDA*, 待会儿会讲, 不过我没用到
- 哪位大神破纪录了跟我吱一声